Verify Application Integrity

The Verify Application Integrity screen offers functions verifying different aspects of the integrity of Natural applications. These functions support quality assurance.

- **Consistency**
  The functions find programs or objects used by programs that have been changed after compilation.
- **Completeness**
  The Objects not implemented but referenced functions find programs or objects that are referenced by programs but are not implemented.
- **Correctness**
  The Objects implemented but not referenced functions find implemented objects that are not referenced.

This section covers the following topics:

- The Verify Application Integrity Menu
- Verify Consistency
- Verify Implementation
- Verify References
- How to Handle Objects that are not Referenced
- Avoiding Dynamic Invocation
- Verifying All Aspects of an Application

The Verify Application Integrity Menu

The Verify Application Integrity menu is displayed with code A in the XRef menu.

```
    13:04:59           *****  P R E D I C T  4.2.2  *****             2002-07-31
    Library: NEWDIC         - Verify Appl. Integrity -      DBnr:   180 Fnr:    54




                   Code Function
                   ---- ----------------------------------------
                     X  Consistency of application
                     I  Objects not implemented but referenced
                     R  Objects implemented but not referenced
                     A  Verify all
                   ---- ----------------------------------------

                Code: ?




   Command ===>
   Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
         Invp  GDAV  Quit  Sets  Rule  Copy  XRef  View  OSet  SPfk  Main  Exit
```


Code	Function	Answer to the Question...	Comments
X	Consistency of application.	Which programs or objects that are used by programs have been modified since compilation	
I	Objects not implemented but referenced	Which programs or objects referenced by programs are not implemented	
R	Objects implemented but not referenced	Which implemented objects are not referenced	
A	Verify all		A second screen is displayed for marking functions to be executed with X.

The different functions are described in the sections below.

Verify Consistency

If work with XRef data is to be effective, the data must be current:

- Source programs must not have been changed after compile time. This ensures both the integrity of source programs and object codes as the consistency of XRef data with both of them.
- All resources used by the program (such as copy code, data areas and processing rules) must not have been changed or deleted after compilation of a program.
  For global data areas, this applies only if the Natural parameter RECAT (dynamic recatalog) is set to OFF. This parameter is explained in the Natural Administrator's documentation.

The Verify Consistency screen is displayed with code X in the Verify Application Integrity menu.

```
    13:06:38            *****  P R E D I C T  4.2.2  *****              2002-07-31
    Library: NEWDIC         -  Verify  Consistency  -       DBnr:   180 Fnr:    54




                   Code Object
                   ---- ----------------------------------------
                    X   Timestamp of source, module and xref data
                    R   Timestamp of resources used in programs
                   ---- ----------------------------------------

               Code: ?
          Save set: N   (Y,N)
```


Code	Function	Answer to the Question...	Comments
X	Timestamp of source, module and xref data	Which programs have been changed after compilation (and generation of XRef data)	If the XRef data is older than the object code, simply recatalog the program. If XRef data exists for which there is no program to recatalog, the data dictionary administrator should delete the XRef data. See Using Predict Special Functions.
R	Timestamp of resources used in programs	Which views, data areas, copy code and free or automatic processing rules have been modified since programs that refer to them were cataloged	

Any inconsistencies can usually be corrected by recataloging the whole application (provided that all sources still exist).

Parameter	
Save set	Any user defined in Predict can save the types and names of programs listed by function R in one or more sets. See Using Sets.

When you enter code X or R, the output screen shows the column S/C/X:

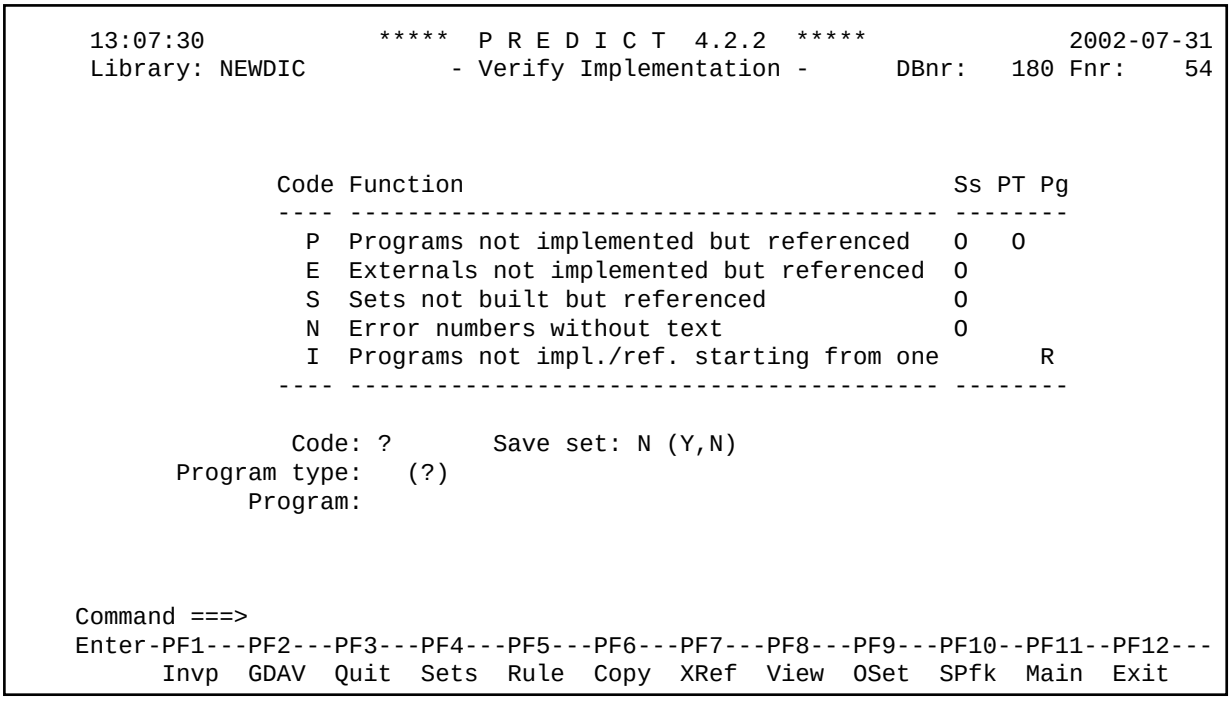
Code	Explantation
S	Source
C	Cataloged
X	XRef data

When you enter code X, the following output is possible in column Note:

Note	Explantation
no XRef data	No XRef data exist on current FDIC file.
XRef invalid	Timestamp of XRef data and cataloged object do not match.
recataloged	Object was recataloged. Timestamp of source and cataloged object do not match.
not cataloged	Source is not cataloged.
source updated	Timestamp of source and cataloged object do not match.

Verify Implementation

All objects referred to should be implemented. The functions of the Verify Implementation menu described below find objects that are referred to but not implemented. The Verify Implementation is displayed with code I in the Verify Application Integrity menu.



Code	Function	Answer to the question...	Comments
P	Programs not implemented but referenced	Which Natural programs that are referred to by programs are not implemented	Natural programs of all types (H, M, N, O, P, S, T, Y and 4) that are invoked but not cataloged are indicated. The invocation method (see Which Invocation Methods are Evaluated) is also indicated. Program type limits this function to programs of a particular type.
E	Externals not implemented but referenced.	Which external entries are referenced but not implemented	An external entry is considered to be implemented if XRef data exists for the 3GL program containing the entry. See the section **Overview of Predict** in the Introduction to Predict documentation for a description where and when XRef data is created for third generation languages. There are three kinds of external entries: - Entries in DBRMs for programs using static SQL - Entries in system programs - Entries in other 3GL programs.
S	Sets not built but referenced	Which retained sets that are referred to by programs (in a FIND ... WITH 'SET' statement or a RELEASE SET 'SET' statement) are not built by any programs (in a FIND ... WITH ... RETAIN AS 'SET' statement)	
N	Error numbers without text	Which error numbers that are referred to by programs (for example, in a REINPUT *<number> statement) have no text assigned in SYSERR	
I	Programs not impl./ref. starting from one	Which objects are referred to, directly or indirectly, by the startup program but are not implemented Which implemented programs are not referenced either directly or indirectly by the startup program	In many applications, all objects that are implemented should be referred to, direct or indirectly, by one particular program: the startup program. The function finds objects that are referred to, directly or indirectly, by the startup program but are not implemented, and all programs that are implemented and not referenced directly or indirectly by the startup program

Parameters		
Save set	Any user defined in Predict can save the types and names of programs listed by function *P, E, S, N* in a set. See Using Sets.	
	All parameters except Save set limit the scope of the function.	
Program type	The type of Natural program for which the function is to be executed. One of the following values can be specified:	
	H	help routine
	M	map
	N	subprogram
	O	command processor
	P	main program
	S	subroutine
	T	dialog
	4	class
	Y	Natural Expert model
	4	class
	*, blank	All
Program	Start program for the function I (Programs not impl./ref. Starting from one).	

Verify References

All objects implemented in the application (except utility programs) should be referred to in the application. The functions of the Verify References menu find objects that are not used.

The Verify References is displayed with code I in the Verify Application Integrity menu.


```
13:13:59            *****  P R E D I C T  4.2.2  *****              2002-07-31
Library: NEWDIC           - Verify References  -        DBnr:   180 Fnr:    54


          Code Function                                  Ss DA DT PT Pg
          ---- ----------------------------------------  --------------
            D  Data areas not referenced                  O     O
            V  Variables in data area not referenced         *  O
            C  Copycodes not referenced
            N  Error numbers not referenced
            P  Programs not referenced                    O        O
            I  Programs not impl./ref. starting from one                R
          ---- ---------------------------------------- --------------
               Code: ?        Save set: N (Y,N)
            Data area:        Data area type :   (?)
      Program type:   (?)
           Program:



Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Invp  GDAV  Quit  Sets  Rule  Copy  XRef  View  OSet  SPfk  Main  Exit
```

Code	Function	Answer to the Question...	Comments
D	Data areas not referenced	Which cataloged data areas stored in the current library are not referred to by any programs	
V	Variables in data area not referenced	Which variables in a data area stored in the current library are not referred to by any programs	
C	Copycodes not referenced	Which copy code stored in the current library is not referred to by any programs	
N	Error numbers not referenced	Which application error numbers that have texts assigned are not referred to by any programs	
P	Programs not referenced	Which programs are not referred to by any other program	
I	Programs not impl./ref. starting from one.	Which objects are referred to, directly or indirectly, by the startup program but are not implemented Which programs are implemented but not referenced either directly or indirectly by the startup program	In many applications, all objects that are implemented should be referred to, direct or indirectly, by one particular program: the startup program.

Parameters		
Save set	Any user defined in Predict can save the types and names of programs listed by function *D* or *P* in a set. See Using Sets.	
Data area	The data area(s) for which the unused variables are to be checked. Asterisk notation is possible. **Note:** Enter the string *DYNAMIC if all area independent variables (AIV) are to be reported. **Note:** Only data areas in the current library will be checked.	
Data area type	Type of data area used by the program. Valid values:	
	G	global data area
	L	local data area
	P	parameter data area
Program type	The type of Natural program for which the function is to be executed. One of the following values can be specified:	
	H	help routine
	M	map
	N	subprogram
	O	command processor
	P	main program
	S	subroutine
	T	dialog
	Y	Natural Expert model
	4	class
	*, blank	All
Program	Start program for the function I (Programs not impl./ref. Starting from one).	

Handling Objects that are not Referenced

The following sections describe how objects that are not used might be processed.

Data areas	Data areas that are not used can usually be deleted.
Variables in data areas	A variable that is not referenced may be used in a redefinition. This possibility should be considered before removing apparently unused variables from data areas.
Copy code	Copy code that are not used can be deleted.
Error numbers	An error number that is referenced dynamically (by assignment to a variable) is always marked not referenced.
Natural programs	A program that is invoked dynamically (by assignment to a variable) is always marked not referenced.

Avoiding Dynamic Invocation

An application is documented more accurately if dynamic invocation is avoided. Dynamic invocation can be replaced by a DECIDE statement, as in the following example:

Replace

```
COMPRESS MNT OBJECT INTO PROG LEAVING NO FETCH PROG
```

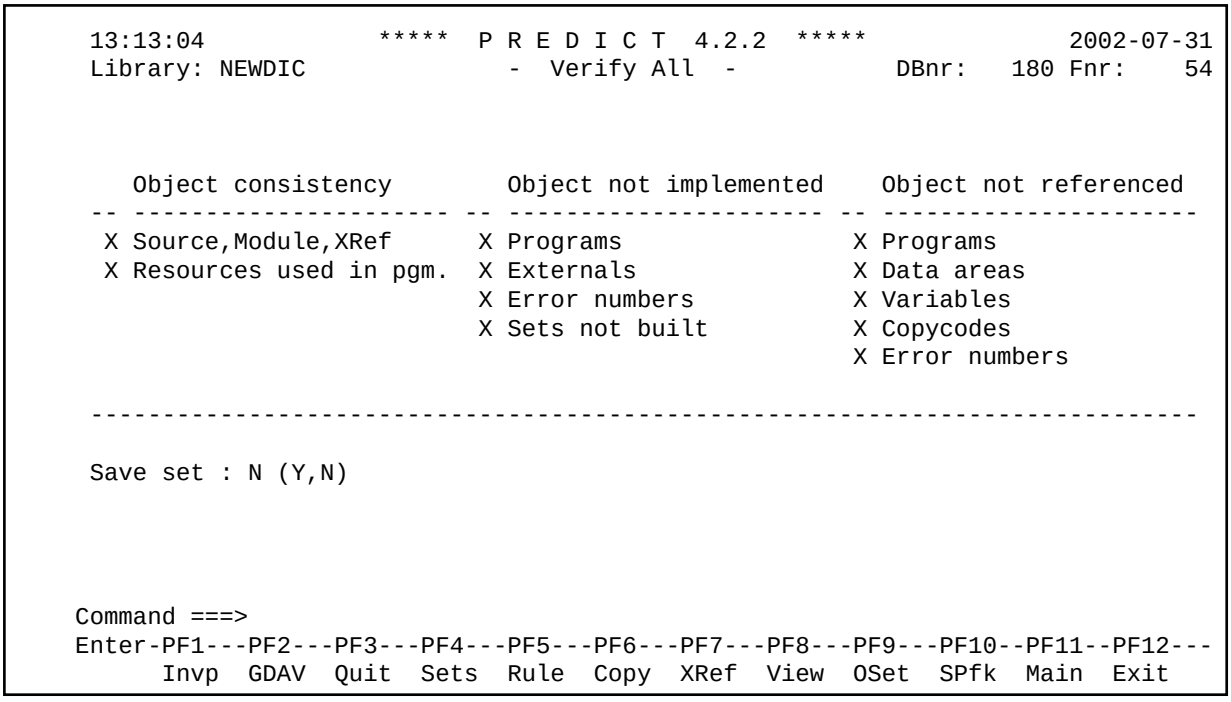
by

```
DECIDE ON FIRST VALUE OF OBJECT
  VALUE EMPL
    FETCH 'MNTEMPL'
  VALUE VEH1
    FETCH 'MNTVEH1'
      .
      .
      .
  NONE VALUE
  WRITE INVALID OBJECT
END-DECIDE
```

Verifying All Aspects of an Application

Before an application is put into production, the functions of the Verify Application Integrity Menu should be executed, their results checked and appropriate action taken to remedy any errors.

Verify All can be used to perform several or all functions of the Verify Application Integrity menu in a single run.



Executing Verify All Interactively

If the verify all function is to be used to call all or several verification functions interactively, the Verify all screen can be displayed with code A in the Verify Application Integrity menu (see the screen on page 3). The functions that are to be executed then have to be marked with X.

Note:
With a large application, Verify all can take a long time. Default parameter settings are taken from the current XREF profile.

Executing Verify All in Batch Mode

Verify all is executed in batch mode with the following command:

VERIFY ALL

Verify All executes the functions selected in the current XREF profile. See the Update user profile screen on #. The default profile is used if no user profile is specified with the following command:

LIST XREF PROFILE user-ID

Saving the Result of Verify All in Sets

A user who is defined in the Predict data dictionary can save the sets of program types and names produced by both Verify consistency functions, four Verify implementation functions and two Verify references functions. A maximum of twelve sets is allowed per user and Natural library. Therefore, not more than four sets should exist in this library for the user when the Verify all function is used with the Save set option Y.

